UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/816,103	03/31/2004	Edward T. Grochowski	42P18305	9899

59796 7590 08/15/2007

INTEL CORPORATION
c/o INTELLEVATE, LLC
P.O. BOX 52050
MINNEAPOLIS, MN 55402

EXAMINER	
FENNEMA, ROBERT E	
ART UNIT	**PAPER NUMBER**
2183	

MAIL DATE	DELIVERY MODE
08/15/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/816,103	GROCHOWSKI ET AL.	
	Examiner	Art Unit	
	Robert E. Fennema	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *6/5/2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-17,20-28,32-51 and 55-71* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-17, 20-28, 32-51, 55-71* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-17, 20-28, 32-51, 55-71 have been considered. Claims 69-71 added as per Applicant's request. Claims 1-5, 7-8, 10, 12, 16-17, 20-21, 23-28, 32, 34-47, 49-51, and 61 amended as per Applicant's request. Claim 18 cancelled as per Applicant's request.

Claim Objections

2. Claim 4 is objected to for referring to a "second" operating system generated thread and second process, when a "first" operating system generated thread and a "first" process have not yet been introduced to the claims (they are mentioned in Claim 3, but Claim 4 is not dependant upon Claim 3).

3. Examiner also notes that Claim 66 is still marked as "new" even though it was introduced in the previous set of claims. Examiner is interpreting Claim 66 as previously presented for the remainder of this action, however, Examiner requests that Applicant confirm the status of Claim 66 in the next response.

4. Examiner further notes that the entire text of Claim 67 appears to be new limitations, however, there is no underlined text to indicate that the subject material was intended to be added into the claim. However, in the interest of furthering prosecution of the case, the Examiner has assumed that the new language in Claim 67 was intended to be added to the claim, as otherwise, the claim would not contain any text.

5. Claim 70 is objected to for concluding in two periods.

Claim Rejections - 35 USC § 102

6. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

7. Claims 1-2, 5-12, 15-17, 20-28 and 32 are rejected under 35 U.S.C. 102(b) as being anticipated by Lawlor et al. (USPN 5,485,626, herein Lawlor).

8. As per Claim 1, Lawlor teaches: A method, comprising:

encountering a non-privileged user-level programming instruction (Column 8, Lines 37-42. Furthermore, the disclosed invention is geared towards not using the operating system (see Column 4, Lines 20-24 and Abstract), meaning that these instructions would be non-privileged, as seen in the Foldoc articles "privileged instruction" and "supervisor mode");

creating, responsive to the programming instruction, a first shared resource thread (shred) that is associated with a first private portion of a first application state (Inherent in multithreaded systems, each thread uses general purpose (among other) registers, which defines their state. The very definition of a thread requires a private state. Column 7, Line 25 discloses that context switching occurs (although not for the

reason mentioned in that paragraph), which is done in multithreading systems to save the private state of each thread when another thread needs to run on that processor) and shares a second shared portion of the first application state with at least a second shred (Column 8, Lines 20-25), wherein creating the first shred is performed without the intervention of an operating system (Abstract and Column 6, Lines 38-42); and

executing, responsive to the programming instruction, the first shred concurrently with at least one of the one or more other shreds (Column 12, Lines 31-32);

wherein creating the first shred is performed in hardware, without the intervention of an operating system (Abstract and Column 6, Lines 38-42).

9. As per Claim 2, Lawlor teaches: The method of claim 1, wherein the first private portion of an application state is associated with at least one of a plurality of registers including general purpose registers, floating point registers, MMX registers, segment registers, a flags register, an instruction pointer, control and status registers, SSE registers, and a MXCSR register (It is inherent that each thread of execution has these registers in their private state, or they cannot function as intended).

10. As per Claim 5, Lawlor teaches: The method of claim 1, wherein the first shred and the second shred share a current privilege level and share a common address translation (Column 8, Lines 37-42 show that the compiler creates instructions to create parallelism (threads/shreds) outside of the operating system, giving them the same privilege level (non privileged) as the others. In addition, Column 8, Lines 20-23

discusses the addressing).

11. As per Claim 6, Lawlor teaches: The method of claim 1, further comprising: receiving a non-privileged user-level programming instruction that encodes a shred destroy operation (Column 29, Lines 27, 49, and Column 30, Line 4).

12. As per Claim 7, Lawlor teaches: The method of claim 1, further comprising communicating between the first shred and the second shred (Column 17, Lines 13-18).

13. As per Claim 8, Lawlor teaches: The method of claim 1, further comprising sharing a system state among at least the first and second shreds (Column 8, Lines 20-28).

14. As per Claim 9, Lawlor teaches: The method of claim 7, wherein said communicating is performed via one or more shared registers (Column 19, Lines 31-32).

15. As per Claim 10, Lawlor teaches: The method of claim 1, further comprising: scheduling, responsive to a user-level programming instruction, the first shred and the second shred without intervention of the operating system (Column 7, Lines 37-42).

16. As per Claim 11, Lawlor teaches: The method of claim 7, wherein:

said communicating is performed via a user-level shred signaling instruction (Column 17, Lines 13-18).

17. As per Claim 12, Lawlor teaches: The method of claim 11, further comprising:

storing at least the first private portion of the first application state in a memory responsive to receipt of a context switch request (inherent in a context switch).

18. As per Claim 15, Lawlor teaches: The method of claim 1, wherein:

the shred is to perform input/output (I/O) operations (Column 12, Lines 35-41).

19. As per Claim 16, Lawlor teaches: The method of claim 1, wherein:

the one or more shreds are to perform computation functions (Column 12, Lines 31-32).

20. As per Claim 17, Lawlor teaches: An apparatus, comprising:

execution resources to execute a plurality of instructions (Column 10, Lines 47 and 51, the processors),

the execution resources to receive a non-privileged user instruction (Column 8, Lines 37-42. Furthermore, the disclosed invention is geared towards not using the operating system (Abstract and Column 6, Lines 38-42), meaning that these instructions would be non-privileged);

the execution resources further to, responsive to the received instruction, execute a first shared resource thread ("shred") concurrently with at least a second shred (Column 12, Lines 31-32); and

a shared register to be addressable by a user-level instruction (Column 17, Line 13, the SRQ object. Column 8, Lines 59-60 show that the objects are stored in registers), the shared register to be directly accessible by at least the first shred and the second shred to provide communication between the first shred and the second shred (Column 19, Lines 31-32).

21. As per Claim 20, Lawlor teaches the apparatus of claim 18, wherein the shared register comprises a first register to enable an operating system or BIOS to enable multithreading architecture extensions for user-level multithreading (Column 18, Lines 13-19).

22. As per Claim 21, Lawlor teaches: The apparatus of claim 17, wherein the shared register comprises a first register to be atomically updated to synchronize data between the first and the second shred (Column 7, Line 62 – Column 8, Line 5).

23. As per Claim 22, Lawlor teaches: The apparatus of claim 17, wherein the execution resources include one or more processor cores capable of executing multiple shreds concurrently (Column 10, Lines 47 and 51).

24. As per Claim 23, Lawlor teaches: The apparatus of claim 17, wherein the shared register is to hold at least a portion of a state shared among the first shred and the second shred (Column 8, Lines 20-24, they share an object space containing counters and queues, which can be implemented in registers as seen in Column 8, Lines 59-60).

25. As per Claim 24, Lawlor teaches: The apparatus of claim 17, wherein the first shred and the second shred share a current privilege level and share a common address translation (Column 8, Lines 37-42 show that the compiler creates instructions to create parallelism (threads/shreds) outside of the operating system, giving them the same privilege level (non privileged) as the others. In addition, Column 8, Lines 20-23 discusses the addressing).

26. As per Claim 25, Lawlor teaches: The apparatus of claim 17, further comprising logic to execute a user-level instruction to create the first shred (Column 8, Lines 37-42).

27. As per Claim 26, Lawlor teaches: The apparatus of claim 17, wherein the shared register comprises a first register to be utilized as a register semaphore to synchronize data between the first and the second shred (Column 12, Lines 42-46).

28. As per Claim 27, Lawlor teaches: The apparatus of claim 17, further comprising a group of register to share a system state among the first shred and the second shred

(Column 8, Lines 20-28).

29. As per Claim 28, Lawlor teaches: The apparatus of claim 17, further comprising a group of registers, which does not include the shared register, to hold a private portion of a state to be associated with the first shred (Inherent in multithreaded systems, each thread uses general purpose (among other) registers, which defines their state. The very definition of a thread requires a private state. Column 7, Line 25 discloses that context switching occurs (although not for the reason mentioned in that paragraph), which is done in multithreading systems to save the private state of each thread when another thread needs to run on that processor).

30. As per Claim 32, Lawlor teaches: The apparatus of claim 17, further comprising:

a user-level exception mechanism to report an exception to the first shred

(Column 18, Lines 21-22).

Claim Rejections - 35 USC § 103

31. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

32. Claims 3-4 are rejected under 35 U.S.C. 103(a) as being unpatentable over Lawlor, in view of Galvin et al. (herein Galvin)

33. As per Claim 3, Lawlor teaches: The method of claim 1, but fails to teach:

wherein the first application state is associated with a first-operating-system-generated thread to execute a first process.

Lawlor teaches a first application state having private and shared portions, but does not teach that the state is associated with an operating system generated thread which executes a process. However, Galvin teaches about processors and threads, in that a process/task contains one or more threads to execute the functionality of the process (see pages 111-115, and that this is the organization of how threads work (threads are part of a process/task, and that threads must be part of a process/task). Galvin also teaches on page 112 that the advantage of having multiple threads operate in a single process is that if one thread blocks, the other threads can run, leading to higher throughput and improved performance. Given these two advantages, one of ordinary skill in the art would have been motivated to use the teachings of Galvin to utilize the threads in processes, as that is how they are required to be organized.

34. As per Claim 4, Galvin teaches: The method of claim 1, further comprising:

creating a second operating system generated thread to execute a second process in response to a privileged non-user level programming instruction, the second

operating system generated thread to be associated with a second application state (Galvin, page 106, more than one process can be created); and

creating a third shred, in response to encountering a second non-privileged user-level programming instruction associated with the second process, wherein the third shred is to be associated with a shared portion and a private portion of the second application state (Galvin, Pages 111-112, each process has threads, so the thread/shred of Claim 1 would be made here too),

wherein the second operating system generated thread and the third shred do not share a portion of the first application state (Galvin, page 108, independent processes).

35. Claims 35-44, 50, and 62 are rejected under 35 U.S.C. 103(a) as being unpatentable over Lawlor and Galvin, further in view of Patterson et al. (herein Patterson).

36. As per Claim 35, Lawlor teaches: An article of manufacture, comprising:

a machine-accessible medium including data that, when accessed by a machine, cause the machine to perform operations comprising,

executing the plurality of threads of execution concurrently on multiple instruction sequencers (Column 12, Lines 31-32), but fails to teach:

executing the threads concurrently in a processor in the machine;

receiving user-level programming instructions to execute a plurality of threads of execution, wherein each of the plurality of threads of execution are to be associated with a private state and to share a system state with an OS-generated thread.

Lawlor teaches user-level instructions to execute a plurality of threads, however, Lawlor is silent towards the threads sharing a system state with an OS-generated thread, although Lawlor does teach of operating systems which support threading. However, Galvin teaches about processes and threads, and that a process (or task) is created by the operating system, and that threads are spawned for each process, with their own private states, but also share a common state (the address space of the process, see Pages 111-115), and these processes are created by the operating system (Page 106). Given that a process is essentially a thread, one could consider a process/task to be an operating-system generated thread, and given the need to know how to implement an OS with threading, without Lawlor providing the specifics and high-level implementation, one of ordinary skill in the art would have been motivated to combine the teachings of Galvin into Lawlor's invention in order to see how to use an OS with the invention properly, and would have incorporated the knowledge of processes into the invention (which are also extremely well known in the art of multithreading), thus the user-level threads created by Lawlor would share a state with the process upon which they are created from.

Furthermore, Lawlor teaches that multiple processors can run multiple threads concurrently, however, does not teach that a single processor can execute multiple threads concurrently. However, the idea of executing multiple threads on a single

processor is extremely well known in the art, as illustrated by the textbook teachings of Patterson, which disclose Simultaneous Multi-Threading (herein SMT), along with it's numerous advantages , for example, increased throughput (see Pages 611-612). Given this advantage, one of ordinary skill in the art at the time the invention was made would have been motivated to have each processor in Lawlor execute multiple threads at the same time, to even further increase the throughput of the system.

37. As per Claim 36, Lawlor teaches: The article of manufacture of claim 35, wherein the operations further comprise:

the private state is associated with at least one of a plurality of registers including general purpose registers, floating point registers, MMX registers, segment registers, a flags register, an instruction pointer, control and status registers, SSE registers, and a MXCSR register (Column 18, Lines 14-18).

38. As per Claim 37, Lawlor teaches: The article of manufacture of claim 35, wherein the private state includes a private portion of an application state (inherent), and wherein the plurality of threads of execution are also to share a shared portion of the application state, the shared portion of the application state to be associated with at least one of a plurality of registers including a control register, a flags register, memory management registers, a local descriptor table register, a task register, debug registers, model specific registers, shared registers, and shred control registers (Column 11, Lines

3-5 show the program context is stored in the SV register 112).

39. As per Claim 38, Lawlor teaches: The article of manufacture of claim 35, wherein the machine-accessible medium further includes data that causes the machine to perform operations comprising:

sharing a state among the plurality of threads of execution; and storing the state in one or more registers (Column 8, Lines 20-24, they share an object space containing counters and queues, which can be implemented in registers as seen in Column 8, Lines 59-60).

40. As per Claim 39, Lawlor teaches: The article of manufacture of claim 35, wherein the plurality of threads of execution share a current privilege level and share a common address translation (Column 8, Lines 37-42 show that the compiler creates instructions to create parallelism (threads/shreds) outside of the operating system, giving them the same privilege level (non privileged) as the others. In addition, Column 8, Lines 20-23 discusses the addressing).

41. As per Claim 40, Lawlor teaches: The article of manufacture of claim 35, wherein the one or more user-level programming instructions include an instruction to create one or more of the plurality of threads of execution (Column 8, Lines 37-42).

42. As per Claim 41, Lawlor teaches: The article of manufacture of claim 35, wherein the machine-accessible medium further includes data that causes the machine to perform operations comprising communicating among the plurality of threads of execution (Column 17, Lines 13-18).

43. As per Claim 42, Lawlor teaches: The article of manufacture of claim 35, wherein the machine-accessible medium further includes data that causes the machine to perform operations comprising sharing a system state among the plurality of threads of execution (Column 8, Lines 20-28).

44. As per Claim 43, Lawlor teaches: The article of manufacture of claim 35, wherein the machine-accessible medium further includes data that causes the machine to perform operations comprising communicating between the plurality of threads of execution via one or more shared registers (Column 19, Lines 31-32).

45. As per Claim 44, Lawlor teaches: The article of manufacture of claim 35, wherein an application program controls the plurality of threads of execution directly, including scheduling of the plurality of threads of execution, and wherein an operating system executed by the multiprocessor schedules the OS-generated thread (Column 2, Lines 21-31).

46. As per Claim 50, Lawlor teaches: The article of manufacture of claim 35, wherein the plurality of threads of execution perform input/output (I/O) functions and computation functions (Column 12, Lines 31-41).

47. As per Claim 62, Lawlor teaches: The article of manufacture of claim 35, wherein the one or more user-level programming instructions include an instruction to destroy one or more of the plurality of shreds (Column 29, Lines 27, 49, and Column 30, Line 4).

48. Claims 45-46 are rejected under 35 U.S.C. 103(a) as being unpatentable over Lawlor and Galvin, further in view of Foldoc.

49. As per Claim 45, Lawlor teaches: The article of manufacture of claim 35, wherein the machine-accessible medium further includes data that causes the machine to perform operations comprising:

associating the plurality of threads of execution with the OS-generated thread (Column 8, Lines 20-24), but fails to teach:

suspending the plurality of threads of execution belonging to the OS-generated thread when a context switch request is received through a single one of the plurality of threads of execution. In Column 18, Lines 13-18, Lawlor teaches that the context is saved when a thread goes inactive, but does not teach that all threads/shreds of a process would be made to be suspended on a context switch. However, Foldoc (Context switch) teaches that on a context switch between processes, the entire

process stops running, and another begins. Given that the reason to multithread is to maximize the shared state as much as possible, it would have been obvious to one of ordinary skill in the art at the time the invention was made to shut down all the shreds associated with the thread/process when it was switched out, so the new process could make use of the advantages of parallelism without the other shreds getting in the way.

50. As per Claim 46, Lawlor teaches: The article of manufacture of claim 45, wherein the machine-accessible medium further includes data that causes the machine to perform operations comprising:

storing one or more threads of execution states associated with the plurality of threads of execution when the context switch request is received (Column 18, Lines 13-18).

51. Claims 13-14 are rejected under 35 U.S.C. 103(a) as being unpatentable over Lawlor and Foldoc, further in view of Golliver et al. (USPN 6,378,067, herein Golliver).

52. As per Claim 13, Lawlor teaches the method of claim 1, but fails to teach:

handling with user-level exception handler code an exception generated during execution of the first shred, without intervention of the operating system.

Lawlor teaches that exceptions can be generated and are dealt with (Column 18, Lines 21-22 for example), but does not explicitly disclose how this occurs, or what deals with the exceptions. Golliver teaches a method to handle multiple exceptions at once by

prioritizing them, and dealing with them in a non-operating system exception handler (Column 3, Line 66 – Column 4, Line 6, and Column 4, Lines 22-25. The operating system is not used to process the exception unless explicitly called, as shown in Column 7, Lines 22-27). Given the need for a method to handle exceptions, and the desire to minimize the operating systems involvement in thread processing in Lawlor's invention, and additionally the need to handle multiple exceptions which could come up when executing multiple shreds concurrently, it would have been obvious to one of ordinary skill in the art at the time the invention was made to use Golliver's exception handler to handle Lawlor's exceptions.

53. As per Claim 14, Golliver teaches: The method of claim 13, further comprising:

receiving the exception from an application program (Column 4, Lines 22-26); and

determining whether to report the exception to the operating system (Column 7, Lines 25-27).

54. Claims 33-34 and 59-61 are rejected under 35 U.S.C. 103(a) as being unpatentable over Lawlor, in view of Golliver.

55. As per Claim 33, Lawlor teaches the apparatus of claim 17, but fails to teach:

an exception mechanism to report an exception to an operating system.

Lawlor teaches that exceptions can be generated and are dealt with (Column 18, Lines 21-22 for example), but does not explicitly disclose how this occurs, or what deals with the exceptions. Golliver teaches a method to handle multiple exceptions at once by prioritizing them, and dealing with them in a non-operating system exception handler (Column 3, Line 66 – Column 4, Line 6, and Column 4, Lines 22-25. The operating system is not used to process the exception unless explicitly called, as shown in Column 7, Lines 22-27). Given the need for a method to handle exceptions, and the desire to minimize the operating systems involvement in thread processing in Lawlor's invention (but still be able to use it when necessary), and additionally the need to handle multiple exceptions which could come up when executing multiple shreds concurrently, it would have been obvious to one of ordinary skill in the art at the time the invention was made to use Golliver's exception handler to handle Lawlor's exceptions.

56. As per Claim 34, Golliver teaches: The apparatus of claim 32, further comprising:

a mechanism to a first exception associated with the first shred and a second exception associated with the second shred (Column 3, Line 66 – Column 4, Line 6, and Column 5, Lines 29-31);

wherein the exception mechanism includes a prioritizer to prioritize the first and the second exceptions (Column 4, Line 1); and

wherein the exception mechanism is further to report only one of the prioritized first and second exceptions at a time to the operating system (Column 7, Lines 23-28

disclose the operating system may be able to be told to handle exceptions, and Column 7, Lines 11-13 show that the faults can be reported one at a time).

57. As per Claim 59, Golliver teaches: The apparatus of claim 32, wherein: the user-level exception mechanism is further to vector to a fixed location in order to allow the shred to service to the exception (Figure 4, and Column 7, Lines 23-28, the user handler set).

58. As per Claim 60, Golliver teaches: The apparatus of claim 32, wherein: the plurality of instructions further include a system call instruction to explicitly invoke an operating system to service to the exception (Column 7, Lines 23-28).

59. As per Claim 61, Golliver teaches: The apparatus of claim 34, wherein said prioritizer employs a round-robin scheme (Column 7, Lines 11-12 and Column 5, Lines 30-32).

60. Claims 47-49 are rejected under 35 U.S.C. 103(a) as being unpatentable over Lawlor and Galvin, further in view of Golliver.

61. As per Claim 47, Lawlor teaches the article of manufacture of claim 35, but fails to teach:

wherein the machine-accessible medium further includes data that causes the machine to perform operations comprising:

reporting one or more exceptions to a first therad of the plurality of threads of execution.

Lawlor teaches exceptions occurring (Column 18 Lines 20-21 for example), but does not teach explicitly what happens during an exception. Golliver teaches a method to handle multiple exceptions at once by prioritizing them, and reporting them to a non-operating system exception handler (Column 3, Line 66 – Column 4, Line 6, and Column 4, Lines 22-25. The operating system is not used to process the exception unless explicitly called, as shown in Column 7, Lines 22-27). Given the need for a method to handle exceptions, and the desire to minimize the operating systems involvement in thread processing in Lawlor's invention (but still be able to use it when necessary), and additionally the need to handle multiple exceptions which could come up when executing multiple shreds concurrently, it would have been obvious to one of ordinary skill in the art at the time the invention was made to use Golliver's exception handler to handle Lawlor's exceptions.

62. As per Claim 48, Golliver teaches: The article of manufacture of claim 47, wherein the machine-accessible medium further includes data that causes the machine to perform operations comprising:

reporting the one or more exceptions from an application program (Column 2, Lines 42-47); and

determining whether to report the one or more exceptions to an operating system (Column 7, Lines 23-27).

63. As per Claim 49, Golliver teaches: The article of manufacture of claim 48, wherein the machine-accessible medium further includes data that causes the machine to perform operations comprising:

prioritized-reporting of the one or more exceptions to the operating system; comprising receiving the one or more exceptions concurrently via different threads of the plurality of threads of execution (Column 5, Lines 30-32); and

servicing one of the one or more exceptions according to the prioritized-reporting while suspending exception processing of other exceptions of the one or more exceptions (Column 7, Lines 11-13).

64. Claims 51, 55-58, and 63-66 are rejected under 35 U.S.C. 103(a) as being unpatentable over Lawlor, in view of Patterson.

65. As per Claim 51, Lawlor teaches: A system, comprising:

a microprocessor implementing an instruction set architecture (ISA) (Column 7, Lines 1-4); and

a memory to store one or more instructions of the ISA (Figure 2, Memory 211 and cache 201) that is to allow user-level multithreading operations (Column 7, Lines 1-4), but fails to teach:

the microprocessor capable of executing multiple OS-generated threads, wherein the microprocessor is also capable of concurrently executing multiple shared resource threads (shreds) associated with one of the OS-generated threads.

Lawlor teaches a multiprocessor system, where each processor is capable of working on a single shred at a time, but does not teach that each individual processor is capable of executing more than one shred at a time. However, the idea of executing multiple threads on a single processor is extremely well known in the art, as illustrated by the textbook teachings of Patterson, which disclose Simultaneous Multi-Threading (herein SMT), along with it's numerous advantages, for example, increased throughput (see Pages 611-612). Given this advantage, one of ordinary skill in the art at the time the invention was made would have been motivated to have each processor in Lawlor execute multiple threads at the same time, to even further increase the throughput of the system.

66. As per Claim 55, Lawlor teaches: A system, comprising:

a microprocessor (Column 10, Line 58, Processor 106), including a plurality of user-level multithreading registers wherein the registers are addressable by one or more user-level instructions in each of a plurality of user-level threads and are to support communication among the user-level threads (Column 19, Lines 31-32); and

memory coupled to the microprocessor, the memory to store the one or more user-level instructions (Column 11, Lines 44-46), but fails to teach:

wherein the microprocessor is further to execute the user-level threads concurrently.

Lawlor teaches a multiprocessor system where each processor can work on a single thread/shred at a time, but does not teach that each individual processor is capable of executing more than one thread/shred at a time. However, the idea of executing multiple threads on a single processor is extremely well known in the art, as illustrated by the textbook teachings of Patterson, which disclose Simultaneous Multi-Threading (herein SMT), along with it's numerous advantages , for example, increased throughput (see Pages 611-612). Given this advantage, one of ordinary skill in the art at the time the invention was made would have been motivated to have each processor in Lawlor execute multiple threads at the same time, to even further increase the throughput of the system.

67. As per Claim 56, Lawlor teaches: The system of claim 55, wherein the plurality of user-level multithreading registers further comprises a plurality of shared shred registers to facilitate communication between a plurality of shreds and to facilitate synchronization between the plurality of shreds (Column 19, Lines 31-32).

68. As per Claim 57, Lawlor teaches: The system of claim 56, wherein the plurality of user-level multithreading registers further comprises a plurality of shred control registers to manage the plurality of shreds (Column 18, Lines 13-18).

69. As per Claim 58, Lawlor teaches The system of claim 57, wherein the microprocessor:

receives programming instructions to execute one or more shreds in accordance with the ISA (Column 8, Lines 37-42);

configures one or more instruction sequencers via the ISA (Column 11, Lines 40-42); and

executes the one or more shreds concurrently (Column 12, Lines 31-32).

70. As per Claim 63, Lawlor teaches: The system of claim 51, wherein the one or more instructions include an instruction to create a shred without intervention of an operating system (Column 4, Lines 20-23).

71. As per Claim 64, Lawlor teaches: The system of claim 51, wherein the one or more instructions include an instruction to destroy a shred without intervention of an operating system (Column 29, Lines 27, 49, and Column 30, Line 4).

72. As per Claim 65, Lawlor teaches: The system of claim 51, wherein: the user-level multi-threading operations include concurrent execution of two or more shreds associated with the same thread (Column 4, Lines 10-15, where it says that a shred (fork) is created when a part of the program is found to be able to be executed in parallel, thus all the shreds are threads of an overall process (the thread in this case)).

73. As per Claim 66, Lawlor teaches: The system of Claim 55, wherein the memory is from a plurality of memory devices including DRAM, flash, and EEPROM (Column 11, Lines 44-46).

74. Claims 67-71 are rejected under 35 U.S.C. 103(a) as being unpatentable over Lawlor, further in view of Galvin.

75. As per Claim 67, Lawlor teaches: An apparatus, comprising:

a processor capable of user-level multithreading including:

a first group of resources to hold a per shared resource thread ("shred") application state for a first shred to be created by a first user-level instruction (Inherent in multithreaded systems, each thread uses general purpose (among other) registers, which defines their state. The very definition of a thread requires a private state. Column 7, Line 25 discloses that context switching occurs (although not for the reason mentioned in that paragraph), which is done in multithreading systems to save the private state of each thread when another thread needs to run on that processor);

a second group of resources to hold a per shred application state for a second shred to be created by a second user-level instruction (Inherent in multithreaded systems, each thread uses general purpose (among other) registers, which defines their state. The very definition of a thread requires a private state. Column 7, Line 25 discloses that context switching occurs (although not for the reason mentioned in that

paragraph), which is done in multithreading systems to save the private state of each thread when another thread needs to run on that processor); and

execution resources to concurrently execute the first shred and the second shred (Column 8, Lines 50-56), but fails to teach:

a third group of resources to be shared by the first shred and the second shred to hold a shared application state, the shared application state to be shared by a privileged-level software entity created thread.

Lawlor does not teach that the threads being executed concurrently share application state with a privileged-level software entity created thread (but does teach that threads can share object space, when the address spaces are shared). Galvin teaches about processes and threads, in that a process/task contains one or more threads to execute the functionality of the process (see pages 111-115), and that this is the organization of how threads work (threads are part of a process/task, and that threads must be part of a process/task). Galvin also teaches on page 112 that the advantage of having multiple threads operate in a single process is that if one thread blocks, the other threads can run, leading to higher throughput and improved performance. Given these two advantages, one of ordinary skill in the art would have been motivated to use the teachings of Galvin to utilize the threads disclosed in Lawlor to parallelize processes, where multiple threads reside in the same process (where all threads in a process share address space).

76. As per Claim 68, Lawlor teaches: The processor of Claim 67, wherein:

the first group of resources, the second group of resources, and the third group of resources include registers (Column 8, Lines 59-60, also see other rejections on why the first and second groups (the private ones) contain registers), and wherein the first user-level instruction and the second user-level instruction are an Instruction Set Architecture instructions to create a shred, which are to be associated with a first user software entity and a second user software entity, respectively (Column 4, Lines 5-9).

77. As per Claim 69, Lawlor teaches: The apparatus of Claim 68, further comprising a hardware re-namer to allocate the first group of registers as private registers and the third group of registers as shared registers based on a bit vector (Column 8, Lines 19-24, the shared registers are determined by the object space, which is specified by the address space, which must be defined as a bit vector. Registers which aren't shared are private).

78. As per Claim 70, Galvin teaches: The apparatus of Claim 67, wherein the application state to be shared by the privileged-level software entity thread is not shared with other privileged-level software entity threads, and wherein the privileged level software entity is an operating system (OS) (Pages 108-109, independent processes).

79. As per Claim 71, Lawlor and Galvin teaches: The apparatus of Claim 67, wherein the second group of resources is a copy of the first group of resources, and wherein the first group of resources includes a combination of resources selected from a group

consisting of general registers, floating point registers, SSE registers, instruction pointer, and flags (Galvin, Page 111), and wherein the third group of resources includes a combination of resources selected from a group consisting of general registers, floating point registers, shared communication registers, flags, memory management registers, address translation tables, privilege levels, and control registers (Column 17, Lines 13-19).

Response to Arguments

80. Regarding Applicants arguments for Independent Claim 17 and some of its dependants, rejected under 102(b), Examiner is not persuaded that the amended claims have overcome the art. The Applicant has argued that Lawlor teaches an object to exchange information between threads, and that not only is this not a register, but is also not directly accessible to each shred. However, Examiner notes Column 8, Lines 59-60, which shows that object storage can be implemented using registers, thus the objects are made of registers, and therefore each object is a register. Since each shred can use an instruction to access the object, it certainly appears to be directly accessible. As a result, Examiner has maintained the rejection on the claims.

81. Regarding Applicants arguments for Claim 1, Applicant has argued that Lawson does not teach that a thread is associated with a first private portion, and a second shared portion of an application state. However, it is inherent that all threads have a private portion, and it is in the definition of a thread that a private portion exists. Every

thread needs some kind of private data, which makes up its state. Additionally, Lawson has taught that parts of these threads are shared amongst each other, both in register to register communication, but also teaches that the object storage portion are shared registers, and thus a shared state. Therefore, Examiner is not at all clear upon what Applicant is attempting to argue when it is said that not only does Lawson not teach these features, but that it would be impermissible to combine with a reference which does so, because the Applicant appears to be arguing against an inherency. The Applicants arguments regarding Galvin are found to not be persuasive for the same reasons as just stated.

82. Regarding Applicants arguments regarding Claim 35, Examiner agrees neither Lawson nor Galvin teach executing multiple threads on a single processor, and has used Patterson to teach this limitation. While Applicant has argued that Patterson does not teach concurrent execution of threads on a single processor, Examiner directs the Applicant to pages 610-611 of Patterson, which teach the extremely well known in the art concept of simultaneous multithreading, where multiple threads can run at the same time on a single processor, because of the use of a superscalar pipeline. Therefore, Applicant's arguments for Claim 35 are found to not be persuasive.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Robert E. Fennema whose telephone number is (571) 272-2748. The examiner can normally be reached on Monday-Friday, 8:00-5:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Robert E Fennema
Examiner
Art Unit 2183

RF

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100